# TECNOLÓGICO NACIONAL DE MÉXICO
# INSTITUTO TECNOLÓGICO DE TIJUANA

## SUBDIRECCIÓN ACADÉMICA
DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN

SEMESTRE:
Agosto - Diciembre 2025

CARRERA:
Ingeniería Informática

MATERIA:
Patrones de Diseño

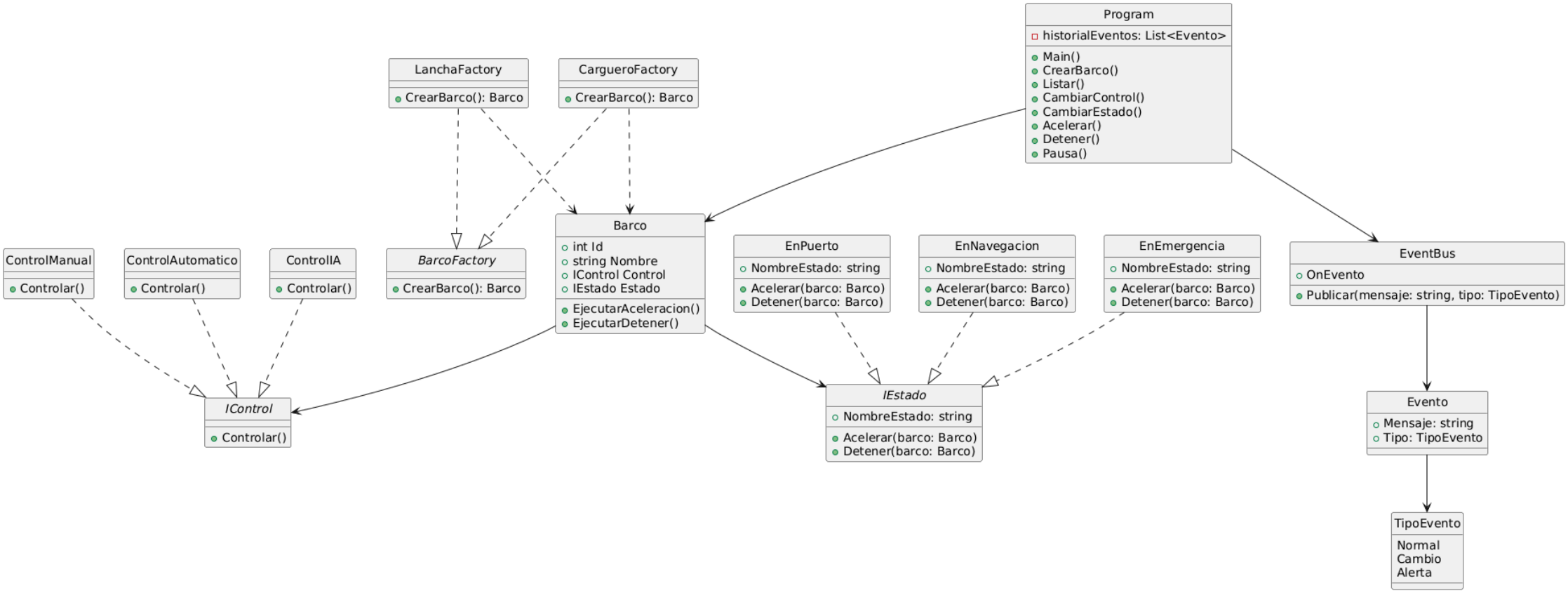
TÍTULO ACTIVIDAD:
Examen unidad 4 y 5

UNIDAD A EVALUAR:
4 y 5

NOMBRE Y NÚMERO DE CONTROL DEL ALUMNO:

Soriano Cabrera Edgar Manuel 21212860

NOMBRE DEL MAESTRO (A):
Maribel Guerrero Luis


Program
- historialEventos: List<Evento>
+ Main()
+ CrearBarco()
+ Listar()
+ CambiarControl()
+ CambiarEstado()
+ Acelerar()
+ Detener()
+ Pausa()
LanchaFactory
+ CrearBarco(): Barco
CargueroFactory
+ CrearBarco(): Barco
BarcoFactory
+ CrearBarco(): Barco
Barco
+ int Id
+ string Nombre
+ IControl Control
+ IEstado Estado
+ EjecutarAceleracion()
+ EjecutarDetener()
ControlManual
+ Controlar()
ControlAutomatico
+ Controlar()
ControlIA
+ Controlar()
IControl
+ Controlar()
EnPuerto
+ NombreEstado: string
+ Acelerar(barco: Barco)
+ Detener(barco: Barco)
EnNavegacion
+ NombreEstado: string
+ Acelerar(barco: Barco)
+ Detener(barco: Barco)
EnEmergencia
+ NombreEstado: string
+ Acelerar(barco: Barco)
+ Detener(barco: Barco)
IEstado
+ NombreEstado: string
+ Acelerar(barco: Barco)
+ Detener(barco: Barco)
EventBus
+ OnEvento
+ Publicar(mensaje: string, tipo: TipoEvento)
Evento
+ Mensaje: string
+ Tipo: TipoEvento
TipoEvento
Normal
Cambio
Alerta

## Factory Method – Creación de barcos

## Class Barco

```
namespace Examen_unidad_4_y_5
{
    17 referencias
    public abstract class Barco
    {
        5 referencias
        public int Id { get; set; }
        16 referencias
        public string Nombre { get; set; }


        5 referencias
        public IControl Control { get; set; }


        9 referencias
        public IEstado Estado { get; set; }

        1 referencia
        public void EjecutarAceleracion() => Estado.Acelerar(this);
        1 referencia
        public void EjecutarDetener() => Estado.Detener(this);

        0 referencias
        public override string ToString()
        {
            return $"[{Id}] {Nombre} | Control: {Control.Nombre} | Estado: {Estado.NombreEstado}";
        }
    }
}
```

## Class Carguero

```
namespace Examen_unidad_4_y_5
{
    2 referencias
    public class Carguero : Barco
    {
        1 referencia
        public Carguero() => Nombre = "Carguero";
    }
}
```

## Class LanchaRapida

```
namespace Examen_unidad_4_y_5
{
    2 referencias
    public class LanchaRapida : Barco
    {
        1 referencia
        public LanchaRapida() => Nombre = "Lancha Rápida";
    }
}
```

## Class BarcoFactory

```
namespace Examen_unidad_4_y_5
{
    3 referencias
    public abstract class BarcoFactory
    {
        3 referencias
        public abstract Barco CrearBarco();
    }

    1 referencia
    public class CargueroFactory : BarcoFactory
    {
        2 referencias
        public override Barco CrearBarco() => new Carguero();
    }

    1 referencia
    public class LanchaFactory : BarcoFactory
    {
        2 referencias
        public override Barco CrearBarco() => new LanchaRapida();
    }
}
```

## Bridge – Modos de control

## Class IControl

```
namespace Examen_unidad_4_y_5
{
    4 referencias
    public interface IControl
    {
        4 referencias
        string Nombre { get; }
        3 referencias
        void Controlar();
    }
}
```

## Class ControlManual

```
namespace Examen_unidad_4_y_5
{
    2 referencias
    public class ControlManual : IControl
    {
        2 referencias
        public string Nombre => "Manual";
        1 referencia
        public void Controlar() => Console.WriteLine("El operador esta controlando manualmente");
    }
}
```

## Class ControlAutomatico

```
namespace Examen_unidad_4_y_5
{
    1 referencia
    public class ControlAutomatico : IControl
    {
        2 referencias
        public string Nombre => "Automatico";
        1 referencia
        public void Controlar() => Console.WriteLine("Sistema automatico navegando");
    }
}
```

## Class ControlIA

```
namespace Examen_unidad_4_y_5
{
    1 referencia
    public class ControlIA : IControl
    {
        2 referencias
        public string Nombre => "IA";
        1 referencia
        public void Controlar() => Console.WriteLine("IA tomando decisiones de navegacion");

    }
}
```

## State – Estados de operación

## Class IEstado

```
namespace Examen_unidad_4_y_5
{
    4 referencias
    public interface IEstado
    {
        6 referencias
        string NombreEstado { get; }
        4 referencias
        void Acelerar(Barco barco);
        4 referencias
        void Detener(Barco barco);
    }
}
```

## Class EnPuerto

```
namespace Examen_unidad_4_y_5
{
    2 referencias
    public class EnPuerto : IEstado
    {
        4 referencias
        public string NombreEstado => "En Puerto";

        2 referencias
        public void Acelerar(Barco barco)
        {
            Console.WriteLine($"El barco {barco.Nombre} no puede acelerar, esta amarrado en puerto");
        }

        2 referencias
        public void Detener(Barco barco)
        {
            Console.WriteLine("El barco ya esta detenido en puerto");
        }
    }
}
```

## Class EnNavegacion

```
namespace Examen_unidad_4_y_5
{
    1 referencia
    public class EnNavegacion : IEstado
    {
        4 referencias
        public string NombreEstado => "En Navegacion";

        2 referencias
        public void Acelerar(Barco barco)
        {
            Console.WriteLine($"{barco.Nombre} acelerando a velocidad de crucero");
        }

        2 referencias
        public void Detener(Barco barco)
        {
            Console.WriteLine($"{barco.Nombre} reduciendo velocidad y deteniendose");
        }
    }
}
```

## Class EnEmergencia

```
namespace Examen_unidad_4_y_5
{
    1 referencia
    public class EnEmergencia : IEstado
    {
        4 referencias
        public string NombreEstado => "En Emergencia";

        2 referencias
        public void Acelerar(Barco barco)
        {
            Console.WriteLine($"Emergencia: {barco.Nombre} solo puede maniobrar, aceleración limitada");
        }

        2 referencias
        public void Detener(Barco barco)
        {
            Console.WriteLine($"{barco.Nombre} no puede detenerse completamente por emergencia");
        }
    }
}
```

## Class Program

```
namespace Examen_unidad_4_y_5
{
    0 referencias
    class Program
    {
        static List<Barco> barcos = new List<Barco>();
        static int contadorId = 1;

        static List<Evento> historialEventos = new List<Evento>();

        0 referencias
        static void Main()
        {

            EventBus.OnEvento += (mensaje, tipo) =>
            {
                historialEventos.Add(new Evento { Mensaje = mensaje, Tipo = tipo });
            };

            while (true)
            {
                Console.Clear();
                MostrarEventosRecientes();

                Console.WriteLine("--- SIMULADOR MARÍTIMO ---");
                Console.WriteLine("1. Crear Barco");
                Console.WriteLine("2. Listar Barcos");
                Console.WriteLine("3. Cambiar Control");
                Console.WriteLine("4. Cambiar Estado");
                Console.WriteLine("5. Acelerar Barco");
                Console.WriteLine("6. Detener Barco");
                Console.WriteLine("0. Salir");

                Console.Write("\nSeleccione una opción: ");
                string opcion = Console.ReadLine();

                switch (opcion)
                {
                    case "1": CrearBarco(); break;
                    case "2": Listar(); break;
                    case "3": CambiarControl(); break;
                    case "4": CambiarEstado(); break;
                    case "5": Acelerar(); break;
```

```
                case "2": Listar(); break;
                case "3": CambiarControl(); break;
                case "4": CambiarEstado(); break;
                case "5": Acelerar(); break;
                case "6": Detener(); break;
                case "0": return;
                default: EventBus.Publicar("Opción inválida", TipoEvento.Alerta); Pausa(); break;
            }
        }
    }


    6 referencias
    static void MostrarEventosRecientes(int max = 5)
    {
        Console.WriteLine("--- Eventos recientes ---");
        var recientes = historialEventos.Skip(Math.Max(0, historialEventos.Count - max));
        foreach (var e in recientes)
        {
            switch (e.Tipo)
            {
                case TipoEvento.Normal: Console.ForegroundColor = ConsoleColor.Green; break;
                case TipoEvento.Cambio: Console.ForegroundColor = ConsoleColor.Yellow; break;
                case TipoEvento.Alerta: Console.ForegroundColor = ConsoleColor.Red; break;
            }
            Console.WriteLine(e.Mensaje);
            Console.ResetColor();
        }
        Console.WriteLine("----------------------\n");
    }

    1 referencia
    static void CrearBarco()
    {
        Console.Clear();
        Console.WriteLine("--- CREAR BARCO ---");
        Console.WriteLine("1. Carguero");
        Console.WriteLine("2. Lancha Rápida");
        Console.Write("Seleccione tipo: ");
        string tipo = Console.ReadLine();

        BarcoFactory factory = null;

        if (tipo == "1") factory = new CargueroFactory();
        else if (tipo == "2") factory = new LanchaFactory();
        else { EventBus.Publicar("Tipo inválido", TipoEvento.Alerta); Pausa(); return; }

        Barco b = factory.CrearBarco();
        b.Id = contadorId++;
```

```
    Barco b = factory.CrearBarco();
    b.Id = contadorId++;
    b.Control = new ControlManual();
    b.Estado = new EnPuerto();

    barcos.Add(b);
    EventBus.Publicar($"Barco creado: {b}", TipoEvento.Normal);
    Pausa();
}

1 referencia
static void Listar()
{
    Console.Clear();
    MostrarEventosRecientes();
    Console.WriteLine("--- LISTA DE BARCOS ---");

    if (barcos.Count == 0)
        Console.WriteLine("No hay barcos registrados.");
    else
        foreach (var b in barcos)
            Console.WriteLine(b);

    Pausa();
}

4 referencias
static Barco Seleccionar()
{
    Console.Write("\nID del barco: ");
    int id;
    if (!int.TryParse(Console.ReadLine(), out id))
    {
        EventBus.Publicar("ID inválido", TipoEvento.Alerta);
        return null;
    }

    var b = barcos.FirstOrDefault(x => x.Id == id);
    if (b == null) EventBus.Publicar("No existe un barco con ese ID", TipoEvento.Alerta);
    return b;
}

1 referencia
static void CambiarControl()
{
    Console.Clear();
    MostrarEventosRecientes();
```

```
1 referencia
static void CambiarControl()
{
    Console.Clear();
    MostrarEventosRecientes();
    Console.WriteLine("--- CAMBIAR CONTROL ---");

    var b = Seleccionar();
    if (b == null) { Pausa(); return; }

    Console.WriteLine("\n1. Manual");
    Console.WriteLine("2. Automático");
    Console.WriteLine("3. IA");
    Console.Write("Seleccione: ");
    string op = Console.ReadLine();

    if (op == "1") { b.Control = new ControlManual(); EventBus.Publicar($"Control de {b.Nombre} cambiado a Manual", TipoEvento.Cambio); }
    else if (op == "2") { b.Control = new ControlAutomatico(); EventBus.Publicar($"Control de {b.Nombre} cambiado a Automático", TipoEvento.Cambio); }
    else if (op == "3") { b.Control = new ControlIA(); EventBus.Publicar($"Control de {b.Nombre} cambiado a IA", TipoEvento.Cambio); }
    else EventBus.Publicar("Opción inválida", TipoEvento.Alerta);

    Pausa();
}

1 referencia
static void CambiarEstado()
{
    Console.Clear();
    MostrarEventosRecientes();
    Console.WriteLine("--- CAMBIAR ESTADO ---");

    var b = Seleccionar();
    if (b == null) { Pausa(); return; }

    Console.WriteLine("\n1. En Puerto");
    Console.WriteLine("2. En Navegación");
    Console.WriteLine("3. En Emergencia");
    Console.Write("Seleccione: ");
    string op = Console.ReadLine();

    if (op == "1") { b.Estado = new EnPuerto(); EventBus.Publicar($"Estado de {b.Nombre} cambiado a En Puerto", TipoEvento.Cambio); }
    else if (op == "2") { b.Estado = new EnNavegacion(); EventBus.Publicar($"Estado de {b.Nombre} cambiado a En Navegación", TipoEvento.Cambio); }
    else if (op == "3") { b.Estado = new EnEmergencia(); EventBus.Publicar($"Estado de {b.Nombre} cambiado a En Emergencia", TipoEvento.Alerta); }
    else EventBus.Publicar("Opción inválida", TipoEvento.Alerta);

    Pausa();
}
```

```
        1 referencia
        static void Acelerar()
        {
            Console.Clear();
            MostrarEventosRecientes();
            Console.WriteLine("--- ACELERAR ---");

            var b = Seleccionar();
            if (b != null)
            {
                b.EjecutarAceleracion();
                EventBus.Publicar($"{b.Nombre} (ID:{b.Id}) aceleró en estado {b.Estado.NombreEstado}", TipoEvento.Normal);
            }

            Pausa();
        }

        1 referencia
        static void Detener()
        {
            Console.Clear();
            MostrarEventosRecientes();
            Console.WriteLine("--- DETENER ---");

            var b = Seleccionar();
            if (b != null)
            {
                b.EjecutarDetener();
                EventBus.Publicar($"{b.Nombre} (ID:{b.Id}) se detuvo en estado {b.Estado.NombreEstado}", TipoEvento.Normal);
            }

            Pausa();
        }

        10 referencias
        static void Pausa()
        {
            Console.WriteLine("\nPresiona ENTER para continuar...");
            Console.ReadLine();
        }
    }

    16 referencias
    public static class EventBus
    {
        public static event Action<string, TipoEvento> OnEvento;
```

```
    16 referencias
    public static class EventBus
    {
        public static event Action<string, TipoEvento> OnEvento;

        15 referencias
        public static void Publicar(string mensaje, TipoEvento tipo)
        {
            if (OnEvento != null)
                OnEvento.Invoke(mensaje, tipo);
        }
    }

    21 referencias
    public enum TipoEvento
    {
        Normal,
        Cambio,
        Alerta
    }

    3 referencias
    public class Evento
    {
        2 referencias
        public string Mensaje { get; set; }
        2 referencias
        public TipoEvento Tipo { get; set; }
    }
}
```

```
--- Eventos recientes ---
Barco creado: [1] Lancha Rápida | Control: Manual | Estado: En Puerto
Barco creado: [2] Carguero | Control: Manual | Estado: En Puerto
------------------------

--- SIMULADOR MARÍTIMO ---
1. Crear Barco
2. Listar Barcos
3. Cambiar Control
4. Cambiar Estado
5. Acelerar Barco
6. Detener Barco
0. Salir

Seleccione una opción:
```

```
--- Eventos recientes ---
Barco creado: [1] Lancha Rápida | Control: Manual | Estado: En Puerto
Barco creado: [2] Carguero | Control: Manual | Estado: En Puerto
------------------------

--- LISTA DE BARCOS ---
[1] Lancha Rápida | Control: Manual | Estado: En Puerto
[2] Carguero | Control: Manual | Estado: En Puerto

Presiona ENTER para continuar...
```

```
--- Eventos recientes ---
Barco creado: [1] Lancha Rápida | Control: Manual | Estado: En Puerto
Barco creado: [2] Carguero | Control: Manual | Estado: En Puerto
Control de Lancha Rápida cambiado a IA
------------------------

--- SIMULADOR MARÍTIMO ---
1. Crear Barco
2. Listar Barcos
3. Cambiar Control
4. Cambiar Estado
5. Acelerar Barco
6. Detener Barco
0. Salir

Seleccione una opción:
```

```
--- Eventos recientes ---
Barco creado: [1] Lancha Rápida | Control: Manual | Estado: En Puerto
Barco creado: [2] Carguero | Control: Manual | Estado: En Puerto
Control de Lancha Rápida cambiado a IA
Estado de Lancha Rápida cambiado a En Emergencia
------------------------

--- SIMULADOR MARÍTIMO ---
1. Crear Barco
2. Listar Barcos
3. Cambiar Control
4. Cambiar Estado
5. Acelerar Barco
6. Detener Barco
0. Salir

Seleccione una opción:
```

```
--- Eventos recientes ---
Barco creado: [1] Lancha Rápida | Control: Manual | Estado: En Puerto
Barco creado: [2] Carguero | Control: Manual | Estado: En Puerto
Control de Lancha Rápida cambiado a IA
Estado de Lancha Rápida cambiado a En Emergencia
------------------------

--- ACELERAR ---

ID del barco: 2
El barco Carguero no puede acelerar, esta amarrado en puerto

Presiona ENTER para continuar...
```

```
--- Eventos recientes ---
Control de Lancha Rápida cambiado a IA
Estado de Lancha Rápida cambiado a En Emergencia
Carguero (ID:2) aceleró en estado En Puerto
Lancha Rápida (ID:1) se detuvo en estado En Emergencia
Carguero (ID:2) se detuvo en estado En Puerto
------------------------

--- DETENER ---

ID del barco: 1
Lancha Rápida no puede detenerse completamente por emergencia

Presiona ENTER para continuar...
```