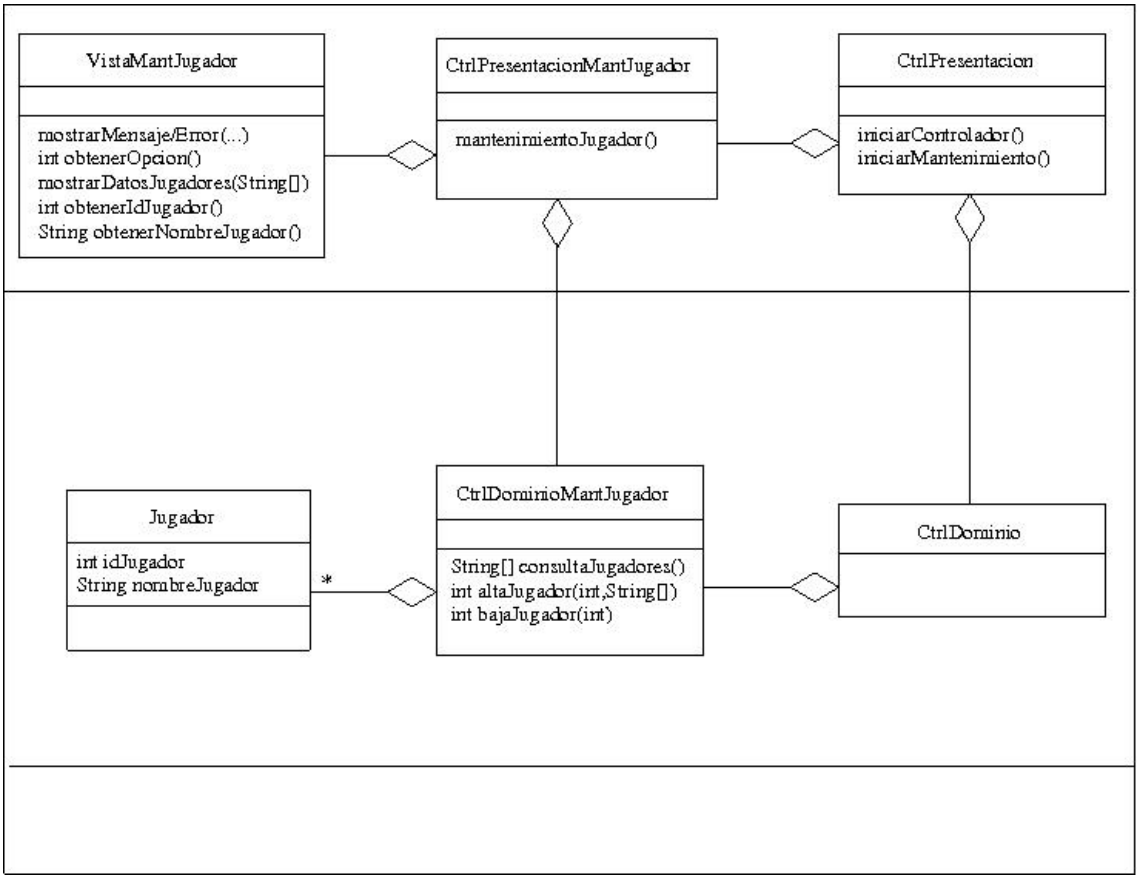
# Ejemplo Implementado de una Arquitectura en 3 Capas: Mantenimiento Jugador



**La clase Jugador**

```
public class Jugador {

  private int idJugador;
  private String nombreJugador;

  public Jugador() {
  }

  public int getId() {
    return idJugador;
  }

  public String getNombre() {
    return nombreJugador;
  }

  public boolean setId (int id) {
    idJugador = id;
    return true;
  }

  public boolean setNombre (String nombre) {
    nombreJugador = nombre;
    return true;
  }

}
```

**La clase CtrlDominio**

```
public class CtrlDominio {

  private CtrlDominioMantJugador CDmj; //*** Agregacion

  public CtrlDominio() {
    CDmj = new CtrlDominioMantJugador();
  }

  public CtrlDominioMantJugador getCtrlDominioMantJugador() {
    return CDmj;
  }

  // Esto es un ejemplo: En un caso real habria mas cosas...
}
```

**La clase CtrlDominioMantJugador**

```
import java.util.*;

public class CtrlDominioMantJugador {

  //*** Los datos se guardan en un TreeMap con clave = idJugador (int)
  private TreeMap<Integer,Jugador> Jugadores; //*** Agregacion

  public CtrlDominioMantJugador() {
    Jugadores = new TreeMap<Integer,Jugador>();
  }

  public Vector<String> consultaJugadores() {
    //*** La informacion se devuelve en estructuras generales (Vector<String>)
    Vector<String> sdatos = new Vector<String>();
    Set setkeys = Jugadores.keySet();
    Iterator iterkeys = setkeys.iterator();
    while (iterkeys.hasNext()) {
      Integer idj = (Integer) iterkeys.next();
      Jugador jug = Jugadores.get(idj);
      String s = "";
      s += jug.getId(); s += " ";
      s += jug.getNombre();
      sdatos.add(s);
    }
    return sdatos;
  }
```

```
  private boolean existeJugador (int idj) { //*** private
    return Jugadores.containsKey(new Integer(idj));
  }




  public int altaJugador (int idj, Vector<String> datos) {

    //*** La informacion viene en estructuras generales (Vector<String>)
    if (existeJugador(idj))
      return 1;
    else //*** Comprobacion de consistencia (innecesaria)
    //*** El identificador esta en la posicion 0
    if (idj != (new Integer(datos.get(0))).intValue())
      return 2;
    else { // Alta del jugador
      Jugador newj = new Jugador();
      if (!newj.setId(idj)) return -1;      // Esto no pasara nunca
      //*** El nombre esta en la posicion 1
      String nomj = datos.get(1);
      if (!newj.setNombre(nomj)) return -1; // Esto no pasara nunca
      Jugadores.put(new Integer(idj),newj);
    }
    return 0;
  }

  public int bajaJugador (int idj) {

    if (!existeJugador(idj))
      return 1;
    else { // Baja del jugador
      Jugadores.remove(new Integer(idj));
    }
    return 0;
  }

}
```

**La clase Main**

```
public class Main {

  private static CtrlPresentacion CP;

  public static void main (String[] args) throws Exception {
    CP = new CtrlPresentacion();
    CP.iniciarControlador();
    // Esto es un ejemplo: Antes de llegar aqui pasara por otros sitios
    CP.iniciarMantenimiento();
  }
}
```

**La clase CtrlPresentacion**

```
public class CtrlPresentacion {

  // Controlador de Dominio
  private CtrlDominio CD;                    //*** Agregacion
  // Controlador de Dominio para el Mantenimiento de Jugador
  // (la agregacion esta en el CD / getCtrlDominioMantJugador / transitividad)
  private CtrlDominioMantJugador CDmj;
  // Controlador de Presentacion para el Mantenimiento de Jugador
  private CtrlPresentacionMantJugador CPmj; //*** Agregacion

  public CtrlPresentacion() {
    CD = new CtrlDominio();
    CDmj = CD.getCtrlDominioMantJugador();
  }

  public void iniciarControlador() throws Exception {
    // En un caso real habra que hacer:
    //  - Inicializar el controlador de dominio
    //  - ...
  }

  public void iniciarMantenimiento() throws Exception {
    CPmj = new CtrlPresentacionMantJugador (CDmj);
    CPmj.mantenimientoJugador();
  }
}
```

**La clase CtrlPresentacionMantJugador**

```
import java.util.*;

public class CtrlPresentacionMantJugador {

  private VistaMantJugador vmj;         //*** Agregacion
  private CtrlDominioMantJugador CDmj;  //*** Agregacion

  CtrlPresentacionMantJugador (CtrlDominioMantJugador c) {
    vmj = new VistaMantJugador();
    CDmj = c; //*** Asigna el controlador de dominio de mantenimiento
    // (podria usar directamente el controlador de dominio, pero habria que
    // duplicar todos los metodos de CtrlDominioMantJugador)
  }

  public void mantenimientoJugador() throws Exception {
    int opcion = -1;
    while (opcion != 0) {
      //*** La vista solo recoge y/o muestra datos
      opcion = vmj.obtenerOpcion();
      switch (opcion) {
        case 0: break;
        case 1: consultaJugadores(); break;
        case 2: altaJugador(); break;
        case 3: bajaJugador(); break;
        default: break;
      }
    }
  }

  private void consultaJugadores() throws Exception { //*** private
    vmj.mostrarMensaje('-',"Consulta de Jugadores");
    //*** El que realmente hace el trabajo es el controlador de dominio
    Vector<String> datos = CDmj.consultaJugadores();
    //*** La vista solo recoge y/o muestra datos
    vmj.mostrarDatosJugadores(datos);
  }
```

```
  private void altaJugador() throws Exception { //*** private
    vmj.mostrarMensaje('-',"Alta de Jugador");
    int idj = vmj.obtenerIdJugador();
    String nomj = vmj.obtenerNombreJugador();
    //*** El que realmente hace el trabajo es el controlador de dominio
    Vector<String> datos = new Vector<String>();
    datos.add((new Integer(idj)).toString());
    datos.add(nomj);
    int codierr = CDmj.altaJugador(idj,datos);
    //*** La vista solo recoge y/o muestra datos
    switch (codierr) {
      case 0:  vmj.mostrarMensaje('-',"Alta efectuada"); break;
      case 1:  vmj.mostrarError("Jugador ya existe"); break;
      case 2:  vmj.mostrarError("Error de inconsistencia"); break;
      default: vmj.mostrarError("Error imposible "+codierr); break;
    }
  }

  private void bajaJugador() throws Exception { //*** private
    vmj.mostrarMensaje('-',"Baja de Jugador");
    int idj = vmj.obtenerIdJugador();
    //*** El que realmente hace el trabajo es el controlador de dominio
    int codierr = CDmj.bajaJugador(idj);
    //*** La vista solo recoge y/o muestra datos
    switch (codierr) {
      case 0:  vmj.mostrarMensaje('-',"Baja efectuada"); break;
      case 1:  vmj.mostrarError("Jugador no existe"); break;
      default: vmj.mostrarError("Error imposible "+codierr); break;
    }
  }

}
```

**La clase VistaMantJugador**

```
import java.util.*;

public class VistaMantJugador {

  private inout io = new inout();
  private int nOpciones = 0;

  ////////////////////////////////////////////////
  // Funciones de visualizacion y opciones del menu
  ////////////////////////////////////////////////

  private void mostrarVista() throws Exception {
    io.writeln("");
    String mensaje = "Menu Mantenimiento Jugador";
    mostrarMensaje('*',mensaje);
    io.writeln("0 - Salir");
    io.writeln("1 - Lista Jugadores"); ++nOpciones;
    io.writeln("2 - Alta"); ++nOpciones;
    io.writeln("3 - Baja"); ++nOpciones;
    io.write("Opcion: ");
  }

  public void mostrarMensaje(char c, String mensaje) throws Exception {
    int n = mensaje.length();
    for (int i=0; i<n; ++i) io.write(c); io.writeln("");
    io.writeln(mensaje);
    for (int i=0; i<n; ++i) io.write(c); io.writeln("");
  }

  public void mostrarError(String mensaje) throws Exception {
    io.writeln(""); io.writeln("ERROR: "+mensaje); io.writeln("");
  }

  public int obtenerOpcion() throws Exception {
    int opcion = -1;
    while (opcion < 0 || opcion > nOpciones) {
      mostrarVista();
      opcion = io.readint();
    }
    io.writeln("");
    return opcion;
  }
```

```
  ////////////////////////////////////////////////////
  // Funciones de visualizacion de datos de los jugadores
  ////////////////////////////////////////////////////

  public void mostrarDatosJugadores (Vector<String> datos) throws Exception {
    int n = datos.size();
    for (int i=0; i<n; ++i)
      io.writeln("Jugador: " + datos.get(i));
  }

  ////////////////////////////////////////////
  // Funciones de obtencion de datos del jugador
  ////////////////////////////////////////////

  public int obtenerIdJugador() throws Exception {
    io.write("Id del jugador: ");
    int idj = io.readint();  String r = io.readline();
    return idj;
  }

  public String obtenerNombreJugador() throws Exception {
    io.write("Nombre del jugador: ");
    String nomj = io.readline();
    return nomj;
  }

}
```

**La clase inout**

Esta clase sólo sirve para la entrada/salida, y es la que hay en la página de la asignatura