MANUAL
DE DISEÑO DEL SISTEMA

1. **Arquitectura de la aplicación**

La aplicación está construida siguiendo las directrices del patrón Modelo-Vista-Controlador. Para adecuarse al patrón, la aplicación se ha dividido en diferentes módulos funcionales**:**

- **Vista**

  La vista principal de la aplicación está construida de forma modular. Las vistas están divididas en:

  - Vista Biblioteca

    La vista biblioteca nos muestra en forma de tabla toda la información referente a la biblioteca, permitiéndonos realizar búsquedas inteligentes, mostrando el resultado en la misma tabla.

  - Vista Reproductor

    En esta vista nos encontramos con los controles básicos del reproductor tales como botones de play, siguiente, anterior, barra de volumen o barra de progreso.

  - Vista Lateral

    En la parte lateral derecha nos encontramos información relativa a las listas de reproducción almacenadas, las canciones que están sonando y el enlace con la biblioteca.

  - Vista Lista Reproducción

  La vista lista de reproducción nos muestra en forma de tabla toda la información referente a la lista actual, permitiéndonos realizar búsquedas inteligentes, mostrando el resultado en la misma tabla.

  Las vistas se comunican con el controlador, según dicta el MVC.

- **Modelo**

El modelo lo forman el conjunto de clases que soportan todas las opciones accesibles desde la vista. Tenemos varias clases principales, encargadas de soportar la lógica de la aplicación:

- Biblioteca Musical
- Control Reproductor
- Lista Reproducción
- Reproductor IS

Además tenemos incluido en el modelo la librería BasicPlayer, encargada de la decodificación de los frames de audio. Esta librería ha sido modificada para optimizarla y mejorar sus prestaciones.

- **Controlador**

  La clase AppController se encarga de hacer las gestiones entre el modelo y la vista.

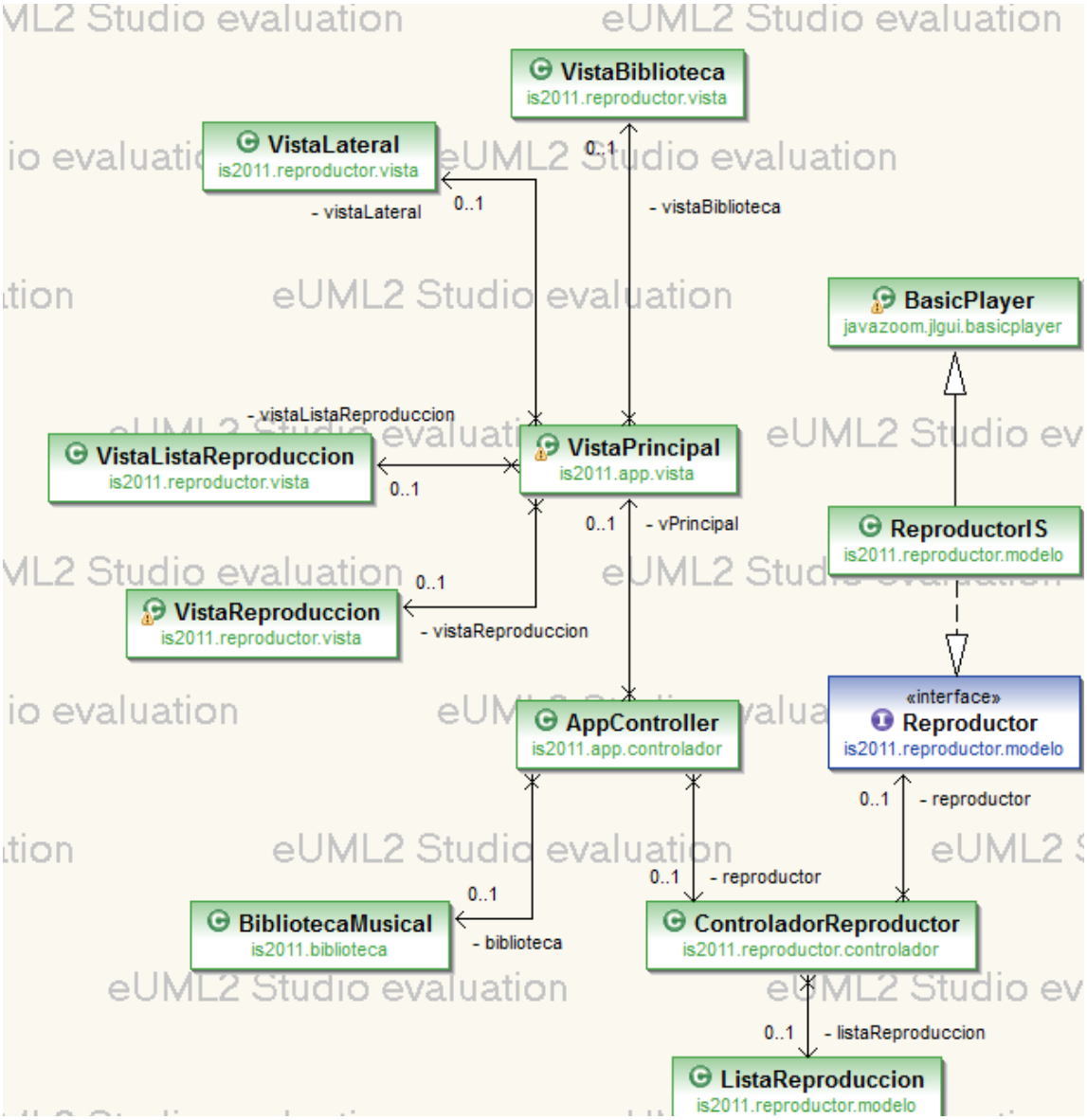
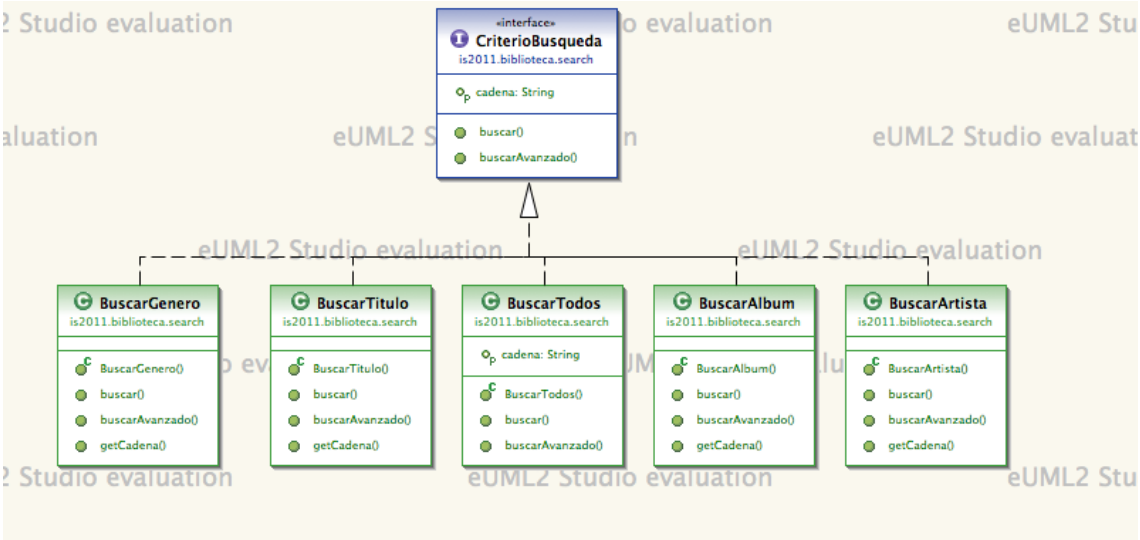


Diagrama de clases reducido mostrando el patrón MVC

2. **Descripción de paquetes**

La arquitectura de la aplicación, como hemos visto en el punto anterior, sigue el patrón MVC. Todo las clases que conforman el MVC están dentro de una estructura jerarquizada, compuesta por paquetes y clases:

- **Is2011**
  - **app**
    - **Controlador**: Paquete en el que nos encontramos con las clases *iAppController.java* y *AppController.java*, que impleme
    - **Preferencias**: Aquí se encuentra la clase *Preferencias.java*, que se encarga de leer y escribir el archivo de preferencias del sistema en formato XML.
    - **Vista**: Paquete donde podemos encontrar la clase *VistaPrincipa.java*, clase que junta el resto de vistas para presentarlas al usuario.
  - **biblioteca**: El paquete biblioteca contiene la clase *BibliotecaMusical.java* y cuatro sub paquetes que contienen los archivos que conforman la biblioteca.
    - **contenedor**: Contiene los contenedores de la biblioteca, a saber, *BibliotecaContainer.java*, que almacena en si la biblioteca, *CancionContainer.java*, que almacena la información de las canciones y *DirectorioContainer.java*, que almacena información relativa a los directorios.
    - **search**: Contiene una clase genérica *CriterioBusqueda.java* que implementan el resto de clases del paquete, dependiendo de que tipo de búsqueda quieras realizar.



- **sort**: Contiene las clases que implementan las ordenaciones, una por cada tipo de ordenación que se permite realizar desde la interfaz. Estas clases son: *sortAlbum.java*, *sortArtista.java*, *sortDuracion.java*, *sortGenero.java* y *sortTitulo.java*.
- **util**: Paquete que contiene clases que implementan las estrategias de actualización de la biblioteca. Implementan el patrón *strategy*. Veremos un diagrama UML en el apartado de patrones.

- **principal**: Contiene a la clase que lanza la aplicación, *Principal.java*, que se encarga de construir los objetos, recuperar datos si estos existieran, mostrar las vistas, etc.
- **reproductor**
  - **controlador**: Paquete que contiene a la clase *ControladorReproductor.java*, clase que se encarga de manejar la playlist y el reproductor. Su diagrama UML se puede observar en el diagrama del MVC.
  - **modelo**: Este paquete contiene a las clases *Reproductor.java*, *ReproductorIS.java* (que

implementa *Reproductor.java* y extiende a la librería BasicPlayer) y *ListaReproduccion.java*

- **listeners**: Paquete que contiene los interfaces ListaReproduccionListener.java y BibliotecaListener.java y las clases BorrarCancionEvent.java y NuevaCancionEvent.java, encargadas de realizar parte de la comunicación entre clases de la aplicación.

▪ **vista**: Paquete que contiene las vistas que conforman la vista principal, *VistaLateral.java, VistaReproductor.java, VistaBiblioteca.java y VistaListaReproduccion.java* . Alguna de estas vistas implementan además a los oyentes de la biblioteca y la lista de reproducción. También encontramos en este paquete la clase *JRoundTextField.java,* usada para hacer el campo de texto de las búsquedas.

o **javazoom**: Paquete que contiene las clases de BasicPlayer.

o **utilidades**: Contiene el fichero RecorreFichero.java, necesario para explorar en las carpetas y subcarpetas del sistema de ficheros. También contiene al paquete estrategias.

▪ **estrategias**: Paquete que contiene clases de utilidad para comprobar que se explorar bien los directorios.

- **Test**: En este paquete nos encontramos los ficheros de prueba de la aplicación. Entre dichos ficheros, encontramos parte de ellos creados con el framework de pruebas JUnit y otros creados de manera similar a JUnit pero sin seguir dicho formato. Adicionalmente nos encontramos con un fichero/log de texto donde escribimos todas y cada una de las pruebas a realizar para comprobar el correcto funcionamiento de la aplicación y seguir la metodología TDD.

- **Recursos**: Paquete que contiene todos los recursos que necesita la aplicación como las imágenes de la vista o el manual de usuario accesible desde la interfaz.

3. **Descripción de SVN**

Para el desarrollo de la aplicación hemos usado una herramienta de control de versiones llamada Subversion (SVN). El servidor que nos lo proporciona es xp-dev (www.xp-dev.com). La estructura de directorios de nuestro SVN es la misma que la detallada en el punto anterior. Más información en la wiki del grupo.

4. **Descripción de ficheros de pruebas**

Como hemos mencionado brevemente en el punto segundo, los ficheros de pruebas que hemos realizado para la aplicación siguen el siguiente patrón:

- Fichero de texto en el que se documenta el diseño de cada caso de prueba. En dicho documento nos encontramos pruebas que luego serán automatizadas y otras que aunque tengan una clase asociada que las testee hay que hacer las comprobaciones de manera manual. También tenemos pruebas que no están escritas en código. Esto es debido a que resulta muy complicado codificar ciertas pruebas, como por ejemplo las pruebas de la interfaz gráfica de usuario.
- Pruebas codificadas
    - Pruebas JUnit: Estos ficheros de prueba están creados bajo el framework de JUnit. Estos ficheros codifican las pruebas de:
        - BibliotecaContainer.java
        - BibliotecaMusical.java
        - DirectorioContainer.java
        - Test de escritura XML.
        - Test sobre el container de las canciones.
    - Pruebas Codificadas no automáticas: Pruebas que no usan el framework de JUnit pero nos han resultado

útiles para testear ciertas funcionalidades y ver que se realizaban de manera correcta:

- Test de actualización de la biblioteca.
- Test sobre la biblioteca musical.
- Test sobre el container de las canciones.
- Test sobre escritura XML.
- Test sobre lectura XML.
- Test sobre la exploración de directorios.
- Test sobre las listas de reproducción.
- Test sobre las preferencias del sistema.

5. **Formato de ficheros XML**

   En la aplicación podemos encontrar tres tipos de ficheros con formato XML:

   - Biblioteca

     El fichero que contiene todos los datos relacionados con la biblioteca tiene el siguiente formato:

```
<biblioteca>
 <directorios>
  <entry>
   <string>J:\WOXTER\MUSICA\BLINK182\Cheshire Cat</string>
   <dir>
    <dirPath>J:\WOXTER\MUSICA\BLINK182\Cheshire Cat</dirPath>
    <listaCanciones>
     <track>
      <trackPath>01 - Carousel.mp3</trackPath>
      <titulo>Carousel</titulo>
      <album>Cheshire Cat</album>
```

```
            <genero>(17)</genero>
            <artista>Blink-182</artista>
            <duracion>195</duracion>
            <pista>1</pista>
          </track>
        </listaCanciones>
      </dir>
    </entry>
  </directorios>
</biblioteca>
```

- Lista de reproducción

  Ficheros que contienen información sobre listas de reproducción. En el figuran las canciones que la conforman y su información como la ruta, el álbum, artista género,... . El formato es el siguiente:

```
<list>
  <track>
    <totalPath>J:\WOXTER\MUSICA\BLINK182\Cheshire Cat\12 - Wasting Time.mp3</totalPath>
    <trackPath>12 - Wasting Time.mp3</trackPath>
    <titulo>Wasting Time</titulo>
    <album>Cheshire Cat</album>
    <genero>(17)</genero>
    <artista>Blink-182</artista>
    <duracion>169</duracion>
    <pista>12</pista>
  </track>
</list>
```

- Preferencias del sistema

  En este archivo guardamos información relativa al estado de la aplicación. De este modo, podemos recuperar las acciones que el usuario haya indicado, como el modo de reproducción, la lista que se carga por defecto, el volumen, posición y tamaño de la ventana de la aplicación o el Look & Feel preferido. El formato es el siguiente:


```
<Preferencias>

<pathPreferencias>C:\Users\carlosmoya\ISPlayer</pathPreferencias>

<pathArchivoPreferenciasSistema>C:\Users\carlosmoya\ISPlayer\ISPlayerPreferences.xml</pathArchivoPreferenciasSistema>

<pathBiblioteca>C:\Users\carlosmoya\ISPlayer\Biblioteca.xml</pathBiblioteca>

<pahtListasDeReproduccion>C:\Users\carlosmoya\ISPlayer\Listas</pahtListasDeReproduccion>

<pathListaReproduccionDefecto>C:\Users\carlosmoya\ISPlayer\Listas\lista1.xml</pathListaReproduccionDefecto>

<pathUltimaLista>C:\Users\carlosmoya\ISPlayer\listaDefecto.xml</pathUltimaLista>
  <posX>546</posX>
  <posY>162</posY>
  <ancho>900</ancho>
  <alto>600</alto>
  <compacta>false</compacta>
  <modoReproduccion>NORMAL</modoReproduccion>
  <volumen>1.0</volumen>
  <abrirUltimaLista>true</abrirUltimaLista>
```

<ultimoDirectorioAbierto>C:\Users\carlosmoya\ISPlayer</ultimoDirectorioAbierto>

<nombreLook>SyntheticaSimple2DLookAndFeel</nombreLook>

<hayCambios>true</hayCambios>

<NOMBRE__DIR>ISPlayer</NOMBRE__DIR>

<DIR__LISTAS>Listas</DIR__LISTAS>

<NOMBRE__PREFERENCIAS>ISPlayerPreferences.xml</NOMBRE__PREFERENCIAS>

<NOMBRE__BIBLIOTECA>Biblioteca.xml</NOMBRE__BIBLIOTECA>

<NOMBRE__LISTA>listaDefecto.xml</NOMBRE__LISTA>

</Preferencias>

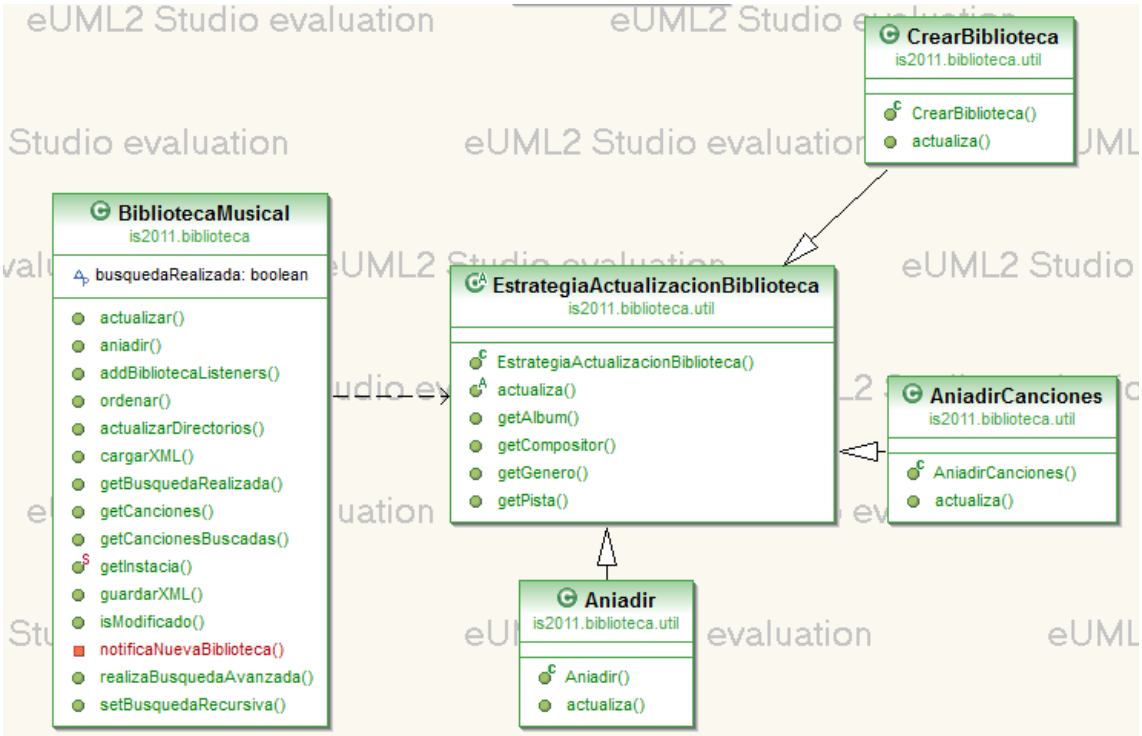
6. Uso de patrones de diseño en la aplicación

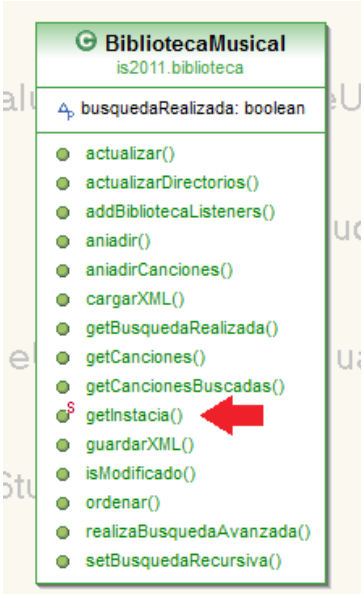
- Estrategia

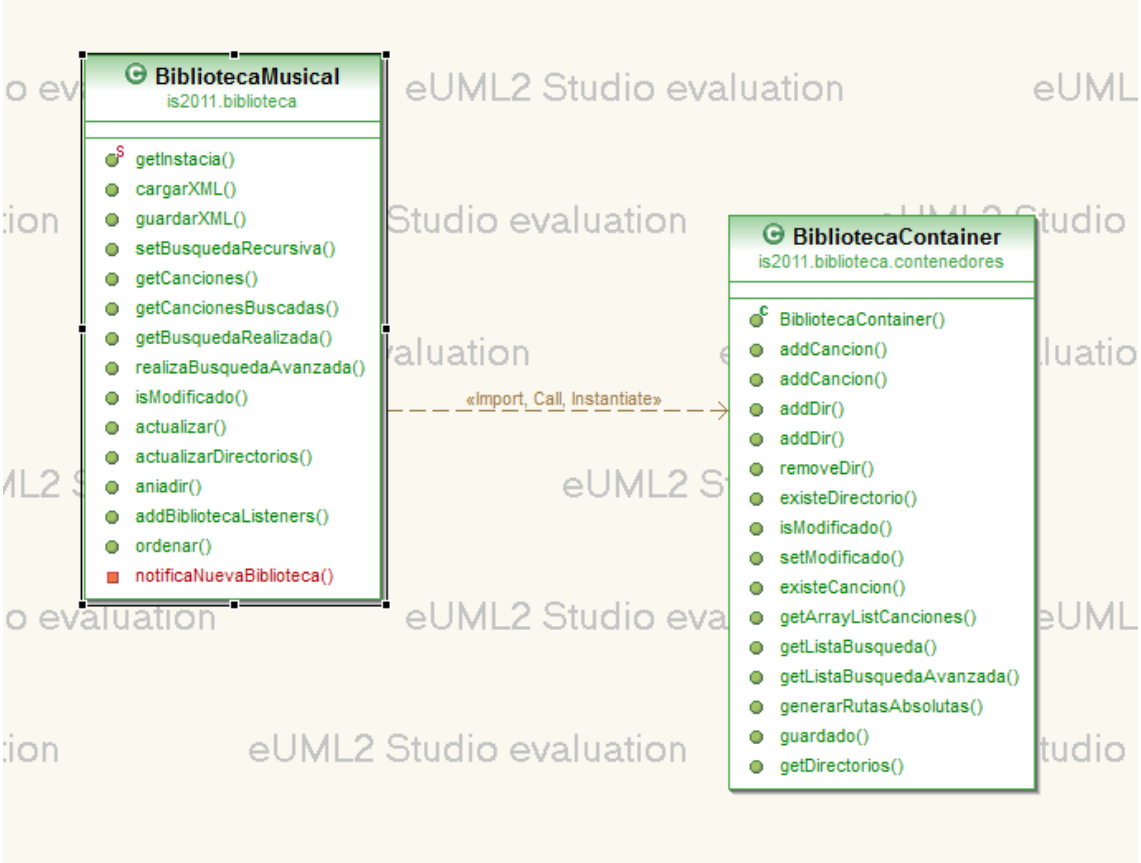
  El patrón estrategia se ha usado como solución a la hora de recorrer ficheros o hacer búsquedas, necesario en la construcción de la biblioteca.

- Singleton

  El patrón singleton se ha usado para la construcción de la biblioteca, ya que por decisiones de diseño pensamos que solo debería existir una.







- Fachada

  Proporciona una interfaz para una o más subclases. En este caso hemos aplicado este patrón para construir la biblioteca.