SEMINARIO

5

Ejercicios de Diseño

Diseño del Software

Grado en Ingeniería Informática del Software

2024-2025

Escuela de Ingeniería Informática
Universidad de Oviedo

César Acebal

Canciones, carpetas y listas de reproducción

Una aplicación de música puede contener

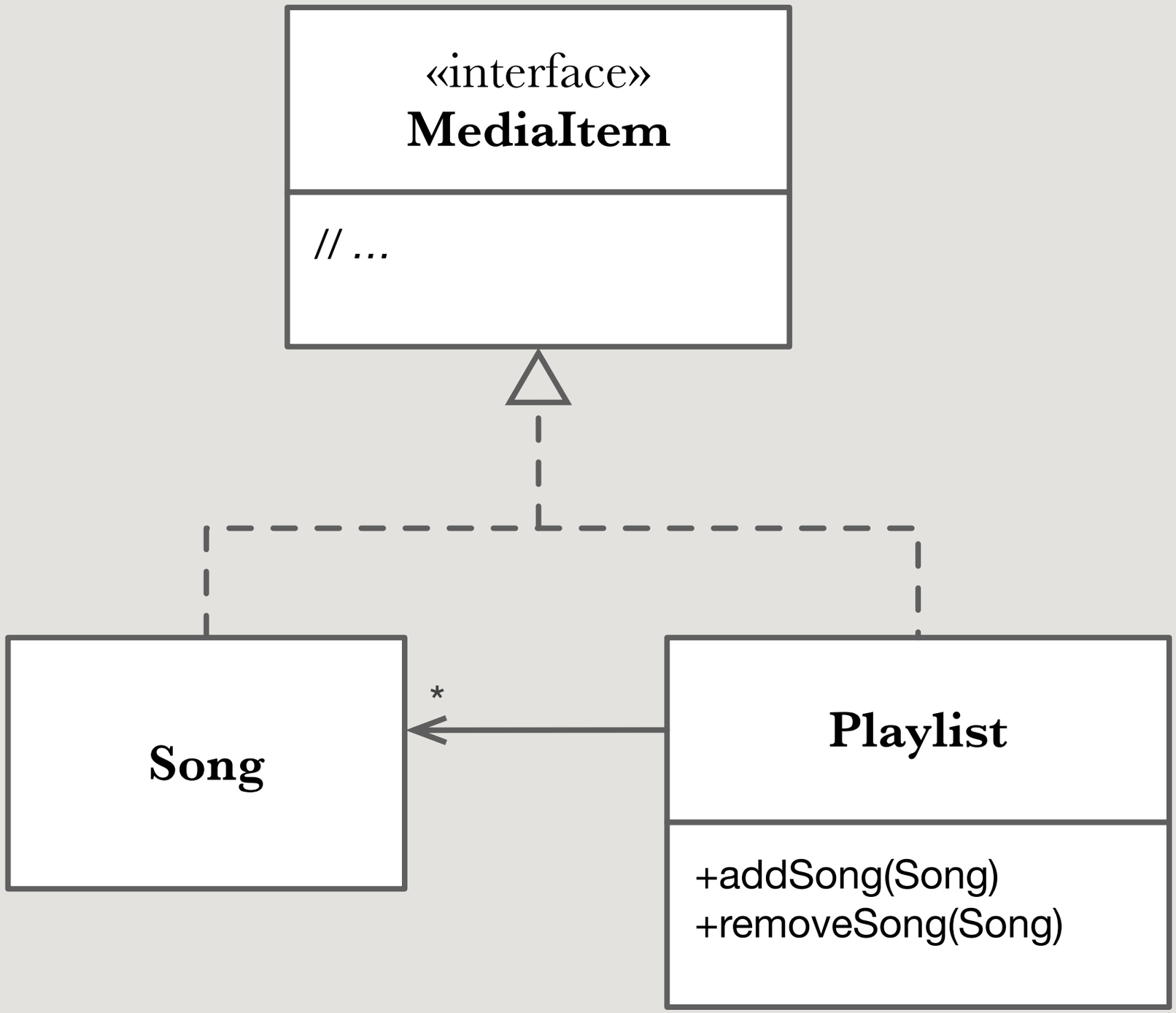
- Canciones
- Listas de reproducción

  Que a su vez solo pueden contener canciones

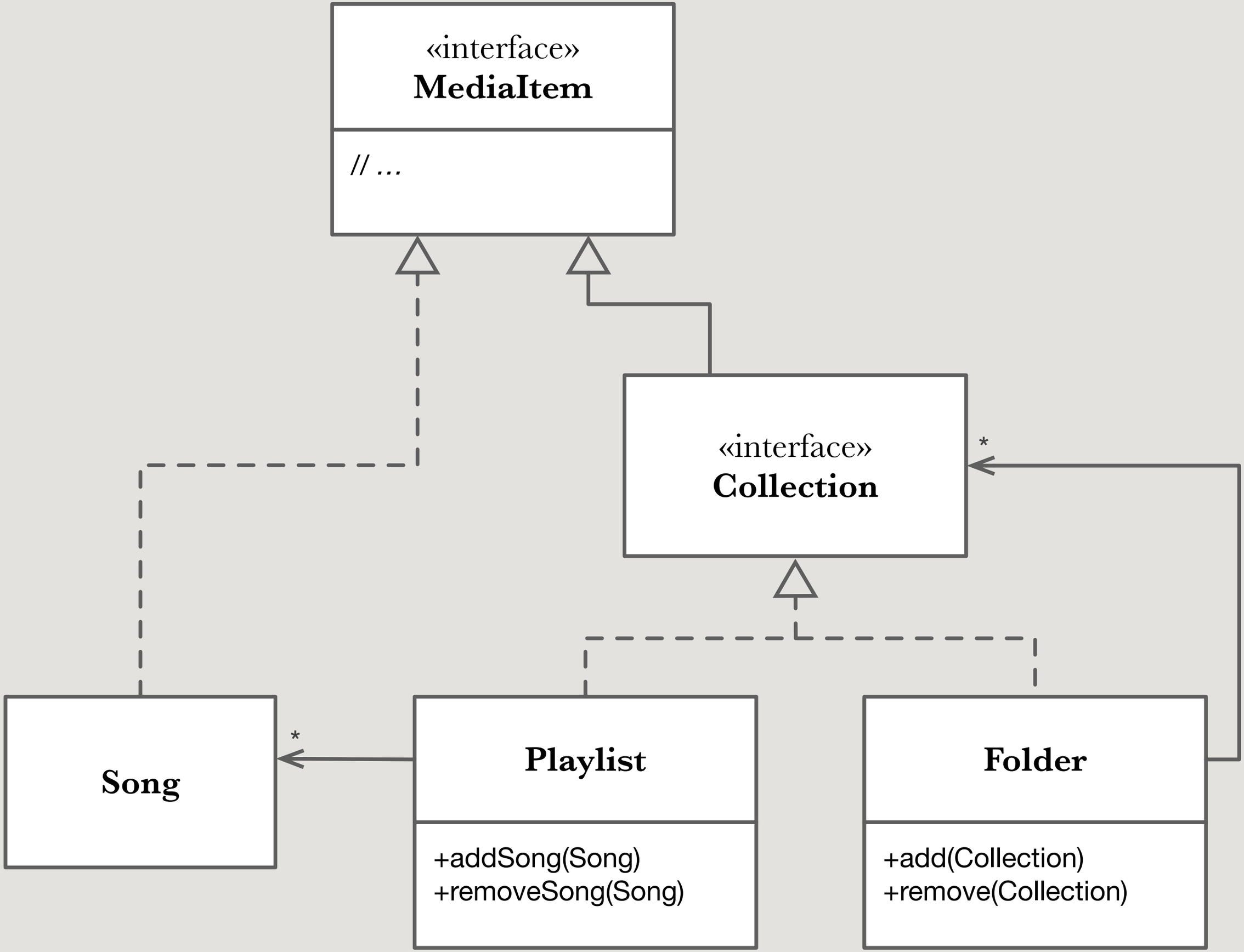
- Carpetas

  Pueden contener listas de reproducción y otras carpetas (pero no canciones sueltas)

¿Cómo se podría modelar lo anterior?



Empecemos representando primero solo
las canciones y las listas de reproducción


«interface»
MediaItem
// …
«interface»
Collection
*
Song
*
Playlist
+addSong(Song)
+removeSong(Song)
Folder
+add(Collection)
+remove(Collection)

Canciones, vídeos y reproductores

Ahora añadimos vídeos a nuestra aplicación multimedia.

Supongamos que las canciones y los vídeos usan reproductores distintos.

¿Cómo podríamos mejorar el siguiente diseño?

Playlist.java

```
public class Playlist {
    // ...
    public void play() {
        for (MediaItem item : items) {
            Player player;
            if (item instanceof Video) {
                player = new VideoPlayer();
            } else if (item instanceof Song) {
                player = new AudioPlayer();
            }
            player.play(item);
        }
    }
}
```

Playlist.java

```java
public class Playlist {
    // ...
    public void play() {
        for (MediaItem item : items) {
            Player player;
            if (item instanceof Video) {
                player = new VideoPlayer();
            } else if (item instanceof Song) {
                player = new AudioPlayer();
            }
            player.play(item);
        }
    }
}
```

¿Qué patrón de diseño es?

Iteradores en la API de Java

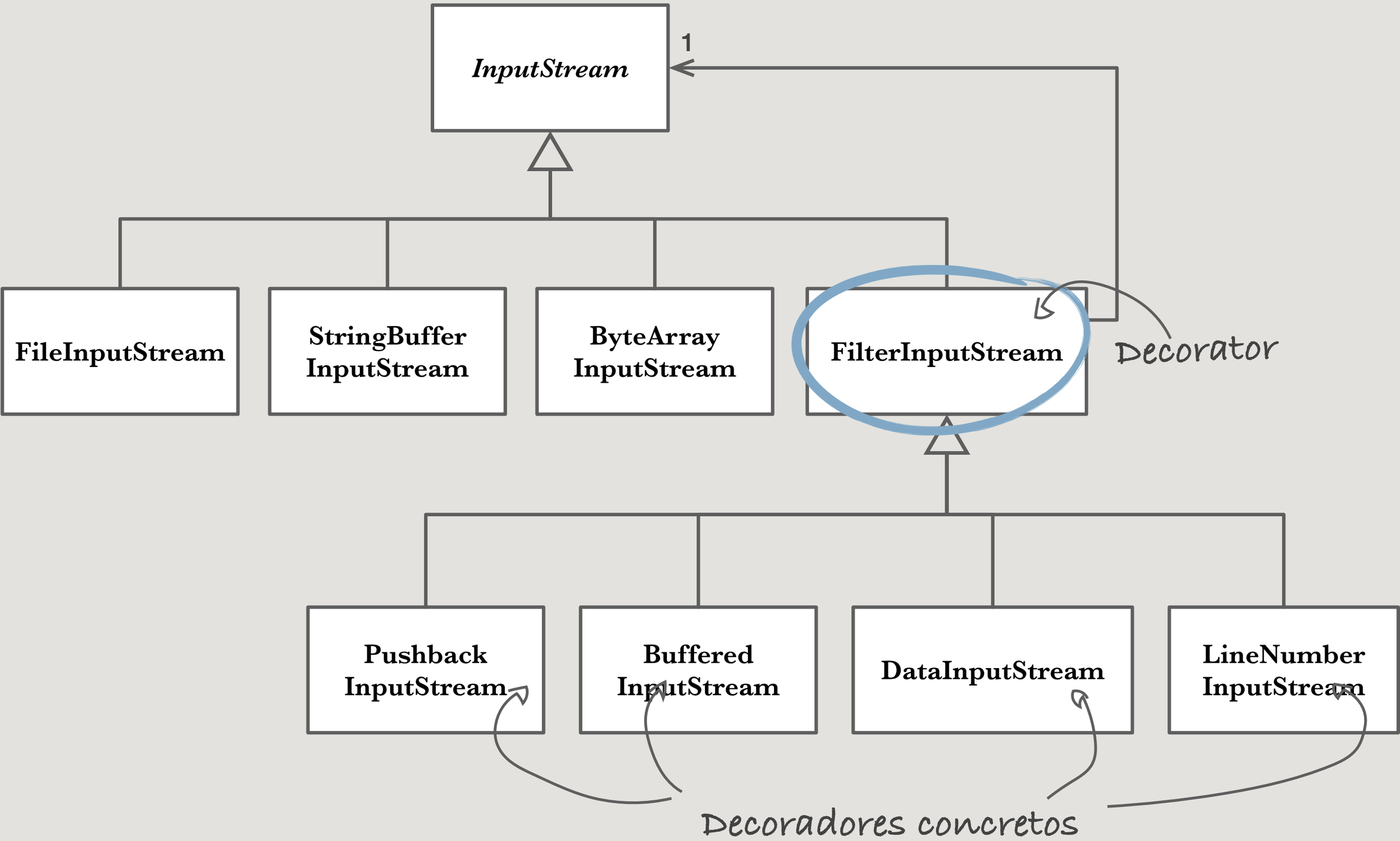
El método iterator de Java Collections, ¿qué patrón de diseño es?

Filtros de la API de Java de E/S

Un ejemplo de uso típico de la API de E/S de Java es:

```
Reader in = new BufferedReader
    (new InputStreamReader(System.in));
```

¿Qué patrón de Java representa?


InputStream
1
FileInputStream
StringBuffer InputStream
ByteArray InputStream
FilterInputStream
Decorator
Pushback InputStream
Buffered InputStream
DataInputStream
LineNumber InputStream
Decoradores concretos

Ejercicio

Escribe un nuevo «input stream» que se integre con el resto de clases de la biblioteca de entrada/salida de Java y que convierta a minúsculas todo lo que lea.

Pruébalo con un pequeño main que lea un fichero de texto y escriba cada carácter leído en la consola.

LowerCaseInputStream.java

```java
public class LowerCaseInputStream extends FilterInputStream {

    public LowerCaseInputStream(InputStream in) {
        super(in);
    }

    public int read() throws IOException {
        int c = super.read();
        return (c == -1 ? c : Character.toLowerCase((char)c));
    }

    public int read(byte[] b, int offset, int len) throws IOException {
        int result = super.read(b, offset, len);
        for (int i = offset; i < offset + result; i++) {
            b[i] = (byte) Character.toLowerCase((char) b[i]);
        }
        return result;
    }
}
```

Main.java

```
// ...

InputStream in = new LowerCaseInputStream(
    new BufferedInputStream(
        new FileInputStream("test.txt")));

int c;
while((c = in.read()) >= 0) {
    System.out.print((char) c);
}

in.close();
```

Framework de pruebas

Un framework de pruebas tiene un método run en su clase TestCase.

Los programadores deben heredar de dicha clase para implementar sus propios tests.

Los creadores del framework quieren que dicho método pueda ejecutar cierto código de inicialización y liberación de recursos antes y después, respectivamente, del método de prueba en sí.

¿Cómo sería?

TestCase.java

```java
public abstract class TestCase implements Test {

    // ...

    public void run() {
        setUp();
        runTest();
        tearDown();
    }

    protected abstract void runTest();
    protected void setUp() {}
    protected void tearDown() {}
}
```

¿Qué patrón de diseño es?